

JavaScript में फ़ंक्शन एक block of code होता है जिसे हम बार-बार उपयोग कर सकते हैं। इसका उद्देश्य एक काम को बार-बार करने के लिए कोड को दोहराने से बचना है।

Function को define कैसे करते हैं?

javascript

Copy

Edit

```
function greet() {  
  console.log("नमस्ते दुनिया!");  
}
```

यहाँ greet नाम का एक function है जो "नमस्ते दुनिया!" प्रिंट करता है।

Function को call कैसे करते हैं?

javascript

Copy

Edit

```
greet(); // यह function को चलाता है
```

Parameters वाले Function:

javascript

Copy

Edit

```
function add(a, b) {  
  return a + b;  
}
```

```
console.log(add(5, 3)); // Output: 8
```

यहाँ a और b parameters हैं और add(5, 3) में 5 और 3 arguments कहलाते हैं।

कुछ महत्वपूर्ण बातें:

Function एक बार define होता है लेकिन कई बार चलाया जा सकता है।

आप function को parameters दे सकते हैं।

Function कुछ return भी कर सकता है।

अगर आप चाहें तो मैं किसी specific function को भी समझा सकता हूँ या कोई real-life example भी दे सकता हूँ।

You said:

Long program

ChatGPT said:

बिल्कुल! नीचे एक लंबा JavaScript प्रोग्राम दिया गया है जो एक Student Management System जैसा है। इसमें हम students की details जोड़ सकते हैं, उन्हें देख सकते हैं और average marks भी निकाल सकते हैं।

मैंने पूरे कोड को हिंदी में comment के साथ समझाया है:

javascript

Copy

Edit

```
// Student Management System
```

```
// Students को store करने के लिए एक array
```

```
let students = [];
```

```
// एक नया student जोड़ने के लिए function
```

```
function addStudent(name, rollNo, marks) {
```

```
  let student = {
```

```
    name: name,
```

```
    rollNo: rollNo,
```

```
    marks: marks
```

```
  };
```

```
  students.push(student); // student को array में जोड़ी
```

```
  console.log(`${name} को सफलतापूर्वक जोड़ा गया!`);
```

```
}
```

```
// सभी students को दिखाने के लिए function
```

```
function showStudents() {
```

```
  console.log("सभी छात्रों की सूची:");
```

```
  students.forEach(function(student, index) {
```

```
    console.log(`${index + 1}. नाम: ${student.name}, रोल नंबर: ${student.rollNo}, अंक:
```

```
    ${student.marks}`);
```

```
  });
```

```
}
```

```
// Average marks निकालने के लिए function
```

```
function calculateAverage() {
```

```
  if (students.length === 0) {
```

```
    console.log("कोई student नहीं है!");
```

```
    return;
```

```
  }
```

```
  let totalMarks = 0;
```

```
  students.forEach(function(student) {
```

```
    totalMarks += student.marks;
```

```
  });
```

```
  let average = totalMarks / students.length;
```

```
  console.log(`औसत अंक: ${average.toFixed(2)}`);
```

```
}
```

```
// Menu जैसी functionality
```

```
function showMenu() {  
  console.log("\n----- Student Management Menu -----");  
  console.log("1. नया छात्र जोड़ें");  
  console.log("2. सभी छात्रों को दिखाएं");  
  console.log("3. औसत अंक निकालें");  
  console.log("4. बाहर निकलें");  
}
```

```
// Main function जो menu के आधार पर काम करेगा
```

```
function main() {  
  let exit = false;  
  
  while (!exit) {  
    showMenu();  
    let choice = prompt("अपना विकल्प दर्ज करें (1-4):");  
  
    switch (choice) {  
      case "1":  
        let name = prompt("छात्र का नाम:");  
        let roll = prompt("रोल नंबर:");  
        let marks = parseFloat(prompt("अंक:"));  
        addStudent(name, roll, marks);  
        break;  
  
      case "2":  
        showStudents();  
        break;  
  
      case "3":  
        calculateAverage();  
        break;  
  
      case "4":  
        console.log("प्रोग्राम समाप्त किया जा रहा है...");  
        exit = true;  
        break;  
  
      default:  
        console.log("कृपया सही विकल्प चुनें।");  
    }  
  }  
}
```

```
// इस function को शुरू करें (यह browser console में चलेगा जहाँ prompt available हो)  
main();
```

यह प्रोग्राम ब्राउज़र में चलाया जा सकता है (जैसे Chrome DevTools में)। prompt() का उपयोग किया गया है इनपुट लेने के लिए।

अगर चाहो तो मैं इसका version Node.js या HTML form के साथ भी बना सकता हूँ। बताओ क्या चाहिए?

You said:

Iska output and parameter argument

ChatGPT said:

चलो अब हम इस प्रोग्राम के output, parameter, और argument को हिंदी में अच्छे से समझते हैं।

1. Parameters और Arguments क्या होते हैं?

Parameters:

Function के अंदर जो variables होते हैं जिन्हें हम define करते हैं, उन्हें parameters कहते हैं।

javascript

Copy

Edit

```
function addStudent(name, rollNo, marks) {  
  // name, rollNo, marks → ये तीनों parameters हैं  
}
```

Arguments:

Function को call करते समय जो actual values भेजते हैं, उन्हें arguments कहते हैं।

javascript

Copy

Edit

```
addStudent("Ravi", 101, 87);  
// "Ravi", 101, 87 → ये तीनों arguments हैं
```

2. इस प्रोग्राम का Output कैसा होगा?

मान लो यूज़र ने कुछ इस तरह input दिए हैं:

पहला student: "Ravi", रोल नंबर: 101, अंक: 87

दूसरा student: "Anita", रोल नंबर: 102, अंक: 93

Output कुछ इस तरह होगा (browser console में):

markdown

Copy

Edit

----- Student Management Menu -----

1. नया छात्र जोड़ें
2. सभी छात्रों को दिखाएं
3. औसत अंक निकालें
4. बाहर निकलें

अपना विकल्प दर्ज करें (1-4): 1

छात्र का नाम: Ravi

रोल नंबर: 101

अंक: 87

Ravi को सफलतापूर्वक जोड़ा गया!

----- Student Management Menu -----

अपना विकल्प दर्ज करें (1-4): 1

छात्र का नाम: Anita

रोल नंबर: 102

अंक: 93

Anita को सफलतापूर्वक जोड़ा गया!

----- Student Management Menu -----

अपना विकल्प दर्ज करें (1-4): 2

सभी छात्रों की सूची:

1. नाम: Ravi, रोल नंबर: 101, अंक: 87

2. नाम: Anita, रोल नंबर: 102, अंक: 93

----- Student Management Menu -----

अपना विकल्प दर्ज करें (1-4): 3

औसत अंक: 90.00

----- Student Management Menu -----

अपना विकल्प दर्ज करें (1-4): 4

प्रोग्राम समाप्त किया जा रहा है...

Recap:

Term	Example	Explanation
Parameter	name, rollNo, marks	Function definition में होते हैं
Argument	"Ravi", 101, 87	Function call में भेजे जाते हैं
Output	Console में जो print होता है	जैसे - "Ravi को सफलतापूर्वक जोड़ा गया!"
अगर तुम चाहो तो मैं इसको HTML form और बटन के साथ भी बना सकता हूँ ताकि real UI में input किया जा सके। बताओ, चाहिए क्या?		

You said:

Function expression

ChatGPT said:

बहुत अच्छा सवाल! अब हम Function Expression को समझते हैं — वो भी सरल हिंदी में।

Function Expression क्या होता है?

JavaScript में Function Expression एक ऐसा तरीका है जिसमें हम एक function को किसी variable में assign करते हैं।

यानि हम function को बना कर किसी नाम (variable) में store कर सकते हैं।

Syntax (रचना)

javascript

Copy

Edit

```
let greet = function() {  
  console.log("नमस्ते दुनिया!");  
};
```

यहाँ पर:

function() एक anonymous function (बिना नाम का function) है।

यह function greet नाम के variable में रखा गया है।

अब हम इस function को इस तरह call कर सकते हैं:

javascript

Copy

Edit

```
greet(); // Output: नमस्ते दुनिया!
```

Example with Parameters and Arguments

javascript

Copy

Edit

```
let add = function(a, b) {  
  return a + b;  
};
```

```
let result = add(5, 3);
```

```
console.log(result); // Output: 8
```

a, b → Parameters

5, 3 → Arguments

add → एक variable जिसमें function रखा गया है।

Function Declaration vs Function Expression

Feature	Function Declaration	Function Expression
---------	----------------------	---------------------

Syntax	function greet() {}	let greet = function() {}
--------	---------------------	---------------------------

Hoisting होता है (ऊपर से चल सकता है) नहीं होता (नीचे से ही चल सकता है)

नाम Named Function Anonymous या Named

Bonus: Arrow Function Expression (ES6)

javascript

Copy

Edit

```
let multiply = (x, y) => x * y;
```

```
console.log(multiply(4, 5)); // Output: 20
```

अगर तुम चाहो तो मैं function expression से एक छोटा real-life project बना सकता हूँ जैसे कि calculator या greeting system। बताओ, करूँ क्या?

You said:

Call apply bind closure long program explain in hindi text english with output

ChatGPT said:

बिलकुल! चलो हम एक लंबा JavaScript प्रोग्राम बनाते हैं जिसमें ये चारों concepts कवर हों:

call

apply

bind

closure

हर एक पार्ट को हम हिंदी में अच्छे से समझाएंगे और साथ में output भी दिखाएंगे।

Full Program with Explanation (Call, Apply, Bind, Closure)

javascript

Copy

Edit

```
// ----- Call, Apply, Bind, Closure Example -----
```

```
// एक object student1
```

```
let student1 = {
```

```
  name: "Ravi",
```

```
  age: 21,
```

```
  greet: function(city, country) {
```

```
    console.log(`Hello, my name is ${this.name}, I am ${this.age} years old. I live in ${city},  
    ${country}.`);
```

```
  }
```

```
};
```

```
// दूसरा object student2
```

```
let student2 = {
```

```
  name: "Anita",
```

```
  age: 22
```

```
};
```

```
// ----- 1. call() -----  
// call method से हम student1 का greet function student2 पर चला सकते हैं  
student1.greet.call(student2, "Delhi", "India");
```

```
// Output:  
// Hello, my name is Anita, I am 22 years old. I live in Delhi, India.
```

```
// ----- 2. apply() -----  
// apply भी call जैसा है, बस arguments array में दिए जाते हैं  
student1.greet.apply(student2, ["Mumbai", "India"]);
```

```
// Output:  
// Hello, my name is Anita, I am 22 years old. I live in Mumbai, India.
```

```
// ----- 3. bind() -----  
// bind एक नया function return करता है जिसे बाद में चलाया जा सकता है  
let boundGreet = student1.greet.bind(student2, "Kolkata", "India");  
boundGreet();
```

```
// Output:  
// Hello, my name is Anita, I am 22 years old. I live in Kolkata, India.
```

```
// ----- 4. Closure Example -----  
function outerFunction(outerVar) {  
  return function innerFunction(innerVar) {  
    console.log(`Outer Variable: ${outerVar}, Inner Variable: ${innerVar}`);  
  };  
}
```

```
// closureFn एक function है जो innerFunction को hold करता है  
let closureFn = outerFunction("JavaScript");  
closureFn("Closure");
```

```
// Output:  
// Outer Variable: JavaScript, Inner Variable: Closure  
हिंदी में समझाइए (English Text + Hindi Explanation):  
1. call()  
call() method किसी function को दूसरे object के साथ चलाने के लिए यूज़ होता है।
```

Syntax: function.call(object, arg1, arg2, ...)

```
javascript  
Copy  
Edit  
student1.greet.call(student2, "Delhi", "India");
```



यहाँ student1.greet को student2 के context में चलाया गया।

## 2. apply()

apply() भी call() जैसा है, लेकिन arguments को एक array में भेजते हैं।

Syntax: function.apply(object, [arg1, arg2, ...])

javascript

Copy

Edit

```
student1.greet.apply(student2, ["Mumbai", "India"]);
```

## 3. bind()

bind() एक नया function return करता है जो दिए गए object के साथ "बांध" दिया जाता है।

Syntax: let newFunc = function.bind(object, arg1, arg2, ...)

javascript

Copy

Edit

```
let boundGreet = student1.greet.bind(student2, "Kolkata", "India");  
boundGreet();
```

## 4. Closure

Closure वह function होता है जो अपने outer function की values को याद रखता है भले ही outer function खत्म हो जाए।

javascript

Copy

Edit

```
function outerFunction(outerVar) {  
  return function innerFunction(innerVar) {  
    console.log(`Outer Variable: ${outerVar}, Inner Variable: ${innerVar}`);  
  };  
}
```

closureFn को outerVar की value याद रहती है क्योंकि innerFunction ने उसे बंद कर लिया है — यही closure है।

Want more?

अगर तुम चाहो तो मैं इसी concept पर एक छोटा UI (HTML + JS) बना सकता हूँ या quiz format में practice questions भी दे सकता हूँ। बताओ क्या चाहिए?

You said:

Ab in chro ka sepretete program banao aur samjaho

ChatGPT said:

बहुत बढ़िया! चलो अब हम Call, Apply, Bind, और Closure के अलग-अलग प्रोग्राम बनाएंगे – हर एक के लिए एक dedicated example और हिंदी में explanation देंगे।

### 1. call() Method

javascript

Copy

Edit

```
let person = {  
  name: "Rahul"  
};
```

```
function sayHello(city) {  
  console.log(`Hello, my name is ${this.name} and I live in ${city}.`);  
}
```

```
sayHello.call(person, "Delhi");
```

Explanation (हिंदी में):

sayHello एक function है जो this.name को access करता है।

हमने call() का इस्तेमाल करके this को person object से जोड़ा।

"Delhi" function का argument है।

Output:

pgsql

Copy

Edit

```
Hello, my name is Rahul and I live in Delhi.
```

### 2. apply() Method

javascript

Copy

Edit

```
let person = {  
  name: "Aarti"  
};
```

```
function introduce(city, country) {  
  console.log(`Hi, I am ${this.name} from ${city}, ${country}.`);  
}
```

```
introduce.apply(person, ["Mumbai", "India"]);
```

Explanation:

apply() भी वही करता है जो call() करता है।

फर्क इतना है कि arguments को array में भेजा जाता है।

Output:

css

Copy

Edit

Hi, I am Aarti from Mumbai, India.

### 3. bind() Method

javascript

Copy

Edit

```
let person = {  
  name: "Neha"  
};
```

```
function greet(language) {  
  console.log(`Namaste! I am ${this.name}, speaking in ${language}.`);  
}
```

```
let greetInHindi = greet.bind(person, "Hindi");  
greetInHindi();
```

Explanation:

bind() एक नया function return करता है जो बाद में use किया जा सकता है।

यहाँ greetInHindi() में this.name हमेशा "Neha" रहेगा।

Output:

css

Copy

Edit

Namaste! I am Neha, speaking in Hindi.

### 4. Closure

javascript

Copy

Edit

```
function outerFunction(name) {  
  let greeting = "Hello";  
  
  return function innerFunction() {  
    console.log(`${greeting}, ${name}!`);  
  };  
}
```

```
let greetUser = outerFunction("Suresh");  
greetUser();
```

Explanation:

outerFunction() एक variable और एक inner function define करता है।

Inner function को return किया जाता है।

Inner function को जब बाद में चलाया जाता है, वो greeting और name को याद रखता है — यही Closure है।

Output:

Copy

Edit

Hello, Suresh!

Recap Table

Concept	क्या करता है?	Syntax	Example
call()	Function को किसी object के साथ तुरंत चलाता है	<code>func.call(obj, arg1)</code>	
apply()	Same as call(), बस args array में	<code>func.apply(obj, [arg1, arg2])</code>	
bind()	Function को object से permanently bind करता है	<code>let newFunc = func.bind(obj)</code>	
Closure	Inner function outer function के variables याद रखता है	<code>return function() { ... }</code>	

अगर तुम चाहो तो मैं इसका एक HTML page के रूप में demo भी बना सकता हूँ — या एक interactive playground जैसे quiz बना सकता हूँ। बोलो क्या पसंद करोगे?

Messages beyond this point are only visible to you

You said:

call backfunction

ChatGPT said:

Callback Function in JavaScript (हिंदी में समझाया गया)

👉 Callback function एक ऐसा function होता है जो किसी दूसरे function को argument के रूप में दिया जाता है, और उसे बाद में call किया जाता है।

📌 इसे हम तब use करते हैं जब हमें कोई काम asynchronous या किसी event के complete होने के बाद करना होता है।

🧠 Callback = "Call me back जब तुम्हारा काम पूरा हो जाए!"

🔧 चलिए एक सिंपल और क्लियर उदाहरण के साथ समझते हैं:

■ Example 1: Basic Callback Function

```
// Step 1: Callback function बनाते हैं function greet(name) { console.log("Hello, " + name); }
```

// Step 2: एक function जो callback को call करता है function processUserInput(callback) { let name = "Ravi"; callback(name); // यहाँ greet() को call किया जा रहा है }

processUserInput(greet);

● Output: Hello, Ravi

📖 Explanation:

greet एक function है जो किसी भी name को greet करता है।

processUserInput में हम greet को callback के रूप में भेजते हैं।

processUserInput के अंदर greet को call किया जाता है।

■ Example 2: Callback with setTimeout (asynchronous)

```
function displayMessage() { console.log("यह message 2 सेकंड बाद दिखेगा"); }
```

```
console.log("Start"); setTimeout(displayMessage, 2000); // 2 सेकंड बाद displayMessage call होगा console.log("End");
```

● Output: Start End (2 सेकंड बाद) यह message 2 सेकंड बाद दिखेगा

📖 Explanation:

setTimeout एक built-in function है जो किसी function को delay से चलाता है।

हमने displayMessage को callback की तरह setTimeout को दे दिया।

■ Example 3: Callback Function with Arguments

```
function calculate(a, b, callback) { let result = a + b; callback(result); // callback को result भेजा }
```

```
function showResult(output) { console.log("Result is: " + output); }
```

```
calculate(5, 10, showResult);
```

● Output: Result is: 15

📖 Explanation:

calculate में दो नंबर जोड़े गए और output callback को भेजा गया।

showResult ने उस output को display किया।

🔙 Summary (सारांश):

Term मतलब

Callback एक function जिसे किसी दूसरे function में call किया जाता है

Asynchronous use Mostly setTimeout, API calls, file read/write में

Advantage Code को modular, readable और flexible बनाता है

अगर आप चाहो तो मैं callback functions से जुड़ी एक mini-project जैसे कि user login simulation या animation भी बना सकता हूँ। बताओ क्या पसंद करोगे?

You said:

code with output

ChatGPT said:

बिल्कुल! अब मैं आपको callback function का एक complete JavaScript code example दूंगा — साथ ही उसका expected output भी दिखाऊंगा। कोड को step-by-step explain भी किया गया है हिंदी में।

✅ Callback Function Code with Output (In JavaScript)

🔗 उदाहरण: हम एक ऐसा function बना रहे हैं जो calculation करता है और result को callback function से output करता है।

javascript

Copy

Edit

// Callback function जो result को display करेगा

```
function showResult(result) {  
  console.log("👉 Calculation Result is:", result);  
}
```

// Main function जो calculation करता है और callback को call करता है

```
function calculate(a, b, callback) {  
  console.log("🧮 Adding numbers:", a, "+", b);  
  let sum = a + b;
```

// अब callback को sum भेजते हैं

```
  callback(sum);  
}
```

// अब calculate function को call करते हैं

```
calculate(10, 15, showResult);
```


🟢 Output:


yaml

Copy

Edit

 Adding numbers: 10 + 15

 Calculation Result is: 25

 दूसरा Example (Asynchronous Callback using setTimeout)

javascript

Copy

Edit

```
function greetUser(name) {  
  console.log("👤 Hello, " + name + "!");  
}
```

```
function loginUser(callback) {  
  console.log("🔑 Logging in...");
```

// 2 सेकंड बाद callback call होगा

```
  setTimeout(function () {  
    console.log("✅ Login successful");  
    callback("Ravi");  
  }, 2000);  
}
```


```
loginUser(greetUser);
```

● Output:


Copy


Edit

 Logging in...

 Login successful

 Hello, Ravi!

 (Note: Output में "Hello, Ravi!" 2 सेकंड की देरी से आता है क्योंकि हमने setTimeout यूज़ किया है।)

 Callback कब use करते हैं?

जब आपको asynchronous काम करना हो (जैसे API call, file reading, user interaction)

Event complete होने के बाद कुछ करना हो

Function को flexible बनाना हो

अगर आप चाहें तो मैं callback के साथ एक real-world mini project बना सकता हूँ — जैसे user registration, या एक custom alert system। बताओ क्या बनाएं?

You said:

curly bracket parenthesis dollar ka use kb krte hai

ChatGPT said:

बहुत अच्छा सवाल! चलिए हम JavaScript (और उसके जैसे modern languages) में इन तीनों symbols का use आसान भाषा में समझते हैं — वो भी clear examples के साथ:

### ✅ 1. Curly Brackets { }

👉 इस्तेमाल: Code block को define करने के लिए कहाँ-कहाँ use होता है?

Functions

Conditions (if/else)

Loops (for, while)

Objects

### 📦 Examples:

#### 1 Function Block:

```
javascript
Copy
Edit
function sayHello() {
  console.log("Hello World");
}
```

#### 2 If Statement:

```
javascript
Copy
Edit
if (5 > 3) {
  console.log("5 is greater than 3");
}
```

#### 3 Object Literal:

```
javascript
Copy
Edit
let person = {
  name: "Ravi",
  age: 25
};
```

👉 याद रखें: Curly Braces {} का मतलब होता है – “Code Block शुरू और खत्म।”



## ✓ 2. Parentheses ( )

👉 इस्तेमाल: Functions को call करने के लिए, parameters देने के लिए, conditions और grouping के लिए  
कहाँ-कहाँ use होता है?

Function declaration और call

Function parameters

if, while, for conditions

Expression grouping

### ■ Examples:

#### 1 Function Declaration:

```
javascript
Copy
Edit
function greet(name) {
  console.log("Hello " + name);
}
```

#### 2 Function Call:

```
javascript
Copy
Edit
greet("Amit");
```

#### 3 If condition:

```
javascript
Copy
Edit
if (a > b) {
  // code
}
```

📌 याद रखें: ( ) का मतलब होता है “condition या function के input”


## ✓ 3. Dollar Sign + Backticks (\${})

👉 इस्तेमाल: Template Literals (string में variables या expressions inject करने के लिए)

### ■ Example:

```
javascript
Copy
Edit
let name = "Neha";
```

```
let greeting = `Hello, ${name}!`;
console.log(greeting);
```

 Output:

Copy

Edit

Hello, Neha!

 यहाँ:

Backticks ( ` ) → Template literal की पहचान

`${ }` → इसके अंदर JavaScript का variable या expression डाल सकते हैं

 दूसरा Example:

javascript

Copy

Edit

```
let a = 5;
let b = 10;
console.log(`Sum of ${a} and ${b} is ${a + b}`);
```

 Output:

csharp

Copy

Edit

Sum of 5 and 10 is 15

 Recap Table

Symbol	इस्तेमाल होता है...	उदाहरण
{ }	Code block, object definition	if, function, object
( )	Parameters, conditions, grouping	function(a, b), if (a > b)
<code>\${ }</code>	Template literal (string interpolation)	Hello, <code>\${name}</code>

अगर आप चाहो तो मैं एक combined code बना सकता हूँ जिसमें इन तीनों का एक साथ उपयोग होता है। बोलो तो दिखाऊँ?

You said:

ek function ka program prxplain kro node.js me run witjhout tweb page me with ourput  
parameter kya hai argument kya hai

ChatGPT said:

बिल्कुल! चलिए हम एक ऐसा JavaScript function बनाते हैं जिसे आप Node.js में चला सकते हैं — यानी बिना किसी वेबपेज (HTML) के — सिर्फ terminal/command prompt से।

हम step-by-step समझेंगे:

Function क्या होता है

Parameters और Arguments क्या होते हैं

Program को Node.js से कैसे चलाएं

Output क्या होगा

✅ Step 1: Function Program (Node.js Compatible)

📁 File name: hello.js

javascript

Copy

Edit

// Function definition with parameters

```
function greetUser(name, age) {  
  console.log(`👋 Hello ${name}, you are ${age} years old.`);  
}
```

// Function call with arguments

```
greetUser("Ravi", 25);
```

✅ Step 2: Parameters vs Arguments

Term मतलब (हिंदी में) Example

Parameter Function define करते समय placeholders name, age (function greetUser(name, age))

Argument Function call करते समय actual value "Ravi", 25 (greetUser("Ravi", 25))

🧠 याद रखो:

Parameters → Variables in function definition

Arguments → Real values when calling the function

✅ Step 3: इस प्रोग्राम को Node.js से कैसे चलाएं?

सबसे पहले ensure करें कि आपने Node.js इंस्टॉल किया है। टर्मिनल में लिखिए:

nginx

Copy

Edit

node -v

ऊपर वाला code एक फ़ाइल में सेव करें, नाम दें: hello.js

अब टर्मिनल में जाकर उस फाइल को रन करें:

nginx

Copy

Edit

node hello.js

● Output in Terminal:

sql

Copy

Edit

👋 Hello Ravi, you are 25 years old.

🔧 Bonus: दूसरा Example with Return

javascript

Copy

Edit

```
function add(a, b) {  
  return a + b;  
}
```

```
let result = add(10, 5);
```

```
console.log("✅ Result is:", result);
```

● Output:

csharp

Copy

Edit

✅ Result is: 15

Summary (सारांश):

Parameters = function में जो variable लेते हो (e.g. name, age)

Arguments = function call करते वक़्त जो value भेजते हो (e.g. "Ravi", 25)

Node.js में JS फाइल टर्मिनल से run कर सकते हो — किसी browser या HTML की ज़रूरत नहीं

अगर आप चाहो तो मैं एक Node.js based छोटा project बना सकता हूँ जैसे calculator, age checker, या input से greet करने वाला function। बताओ क्या बनाऊँ?

You said:

arrow function ka bhi aise he kro

ChatGPT said:

बिलकुल! अब हम Arrow Function का example भी उसी तरह से बनाएंगे:

Node.js में चलने वाला JavaScript code

Parameters और Arguments की समझ

Output के साथ

Step-by-step समझाया गया

✅ Arrow Function Program (Node.js Compatible)

📄 File name: arrowFunction.js

javascript

Copy

Edit

// Arrow Function Definition with Parameters

```
const greetUser = (name, age) => {  
  console.log(`👋 Hello ${name}, you are ${age} years old.`);  
};
```

// Function Call with Arguments

```
greetUser("Neha", 22);
```

✅ Parameters vs Arguments (Same Concept)

Term मतलब (हिंदी में) Example

Parameters Function define करते समय placeholders name, age

Arguments Function call करते समय दिए गए actual values "Neha", 22

🧠 Example में:

Parameters → name, age

Arguments → "Neha", 22

✅ Step-by-step चलाना (Node.js से)

Code को arrowFunction.js नाम की फाइल में सेव करें

टर्मिनल/कमांड लाइन खोलें

वहां ये command चलाएं:

bash

Copy

Edit

node arrowFunction.js

🟢 Output:

sql

Copy

Edit

👋 Hello Neha, you are 22 years old.

✅ Arrow Function: Return के साथ Example

javascript

Copy

Edit

// Arrow function with return

```
const add = (a, b) => a + b;
```

```
let result = add(5, 7);
```

```
console.log("✅ Sum is:", result);
```

🟢 Output:

csharp

Copy

Edit

✅ Sum is: 12

✅ Summary Table

Feature	Traditional Function	Arrow Function
Define	function greet(name) { }	const greet = (name) => { }
Uses this	Own this context	Lexical this (inherits from parent)
Syntax	Verbose	Short & clean
Return	Must write return	Implicit return (if one line)